

Humboldt-Universität zu Berlin

Institut für Informatik



Studienarbeit

Komposition von Web Services

Lars Münzberg

29. Juni 2003

Betreuer: Dipl.-Inf. Axel Martens

Inhaltsverzeichnis

1	Einleitung	4
2	Web Services	6
2.1	Motivation	6
2.2	Problemstellung	7
2.3	Akteure im Web-Service-Ansatz	8
2.4	Abstraktion auf WSFL	9
3	Entwicklung des Kriteriums	11
3.1	Überführung von WSFL in Prozessalgebra	11
3.1.1	Prozessalgebra CCS genügt der Betrachtung	11
3.1.2	Kommunikation ist Synchronisation	12
3.1.3	Abbildung von Senden und Empfangen	13
3.2	Kombinierbarkeitseigenschaft	14
3.2.1	Kombinierbarkeit von Agenten	14
3.2.2	Anforderungen	16
3.3	Definition des Kombinierbarkeitskriteriums	17
4	Beweis	21
4.1	Korrektheit (\Leftarrow)	21
4.2	Vollständigkeit (\Rightarrow)	22
5	Zusammenfassung und Ausblick	24

Abbildungsverzeichnis

1	The 2002 Hype Cycle of Emerging Technologies	7
2	Rollen im Web-Service-Ansatz	9
3	Graph eines Agenten	14
4	verklemmendes komponiertes System	15
5	verklemmungsfreies komponiertes System	15

Verteilte Systeme haben im letzten Jahrzehnt stetig an Bedeutung gewonnen, so sind sie aus dem täglichen Leben nicht mehr weg zu denken. Ein Beispiel hierfür ist das Bezahlen mit der EC-Karte oder die Buchbestellung über das Internet. Die Web-Service-Softwarearchitektur soll die Grundlage einer neuen zukunftssträchtigen Generation verteilter Systeme bilden. Momentan befindet sich die Technologie aber noch in der Entwicklungsphase, unter anderem ist die Komponierbarkeit von Web Services eine offene Frage. In dieser Arbeit soll ein Ansatz zur Entscheidung der als problematisch geltenden semantischen Kompatibilität entworfen werden.

1 Einleitung

Zusehens verstärkt sich die Tendenz, den globaler werdenden Wettbewerb durch globale Zusammenarbeit zu begegnen, in diesem Prozeß werden die lokalen Geschäftsprozesse zu verteilten Geschäftsprozessen komponiert. Dem Internet kommt bei der Komposition von Geschäftsprozesse eine Schlüsselrolle zu, da es sich von einem standardisierten und überall verfügbaren Kommunikationsmedium zu einer Umgebung für verteilte Systeme weiterentwickelt hat. In diesem Entwicklungsprozess haben führende Softwareunternehmen eine als revolutionär bezeichnete Technologie entworfen, mit der lokale Funktionen eines Geschäftsprozesses gekapselt, veröffentlicht, gefunden und zu verteilten Geschäftsprozessen kombiniert werden können, den Web Services.

Der Ausgangspunkt der vorliegenden Arbeit ist die 2001 von IBM [Ley01] spezifizierte *Web Service Flow Language* (WSFL), sie definiert abstrakte Prozesse von Web Services. Ein Web Service besteht aus Aktivitäten, zwischen diesen Aktivitäten bestehen kausale Zusammenhänge. Diese Zusammenhänge werden durch Kontrollstrukturen dargestellt, welche als sequenzielle, alternative und parallele Ausführung der Aktivitäten ausgeprägt sind. Somit bestimmen nicht nur die Aktivitäten sondern auch die Kausalitäten das Verhalten eines Web Services. Diese Punkte vereint ein abstrakter Prozeß, den wir im folgenden als *Web-Service-Prozessmodell* bezeichnen werden. Neben diesem inneren Aufbau besitzt jedes Web-Service-Prozessmodell eine Schnittstelle zur Umgebung, hierdurch wird die Interaktion mit anderen Web Services ermöglicht.

Bei der Komposition zweier Web-Service-Prozessmodelle muss für die fehlerfreie Abarbeitung des entstehenden verteilten Systems darauf geachtet werden,

dass die Web-Service-Prozessmodelle komponierbar sind. Das heißt, es muss sowohl die syntaktische als auch die semantische Kompatibilität erfüllt sein. Syntaktische Kompatibilität wird auf Basis der Schnittstellen entschieden, dies kann leicht durch Parsen geschehen. Die semantische Kompatibilität, also die verklemmungsfreie Zusammenarbeit der beiden Web-Service-Prozessmodelle, muss hingegen auf WSFL entschieden werden. Es existieren dafür momentan keine nennenswerten Ansätze, weshalb es in der hier vorliegenden Arbeit thematisiert wird.

WSFL ist eine Sprache für die Modellierung von Web Services, es fehlt allerdings eine formale Semantik. Diese ist zwingend notwendig für die gewünschte Entscheidung der semantischen Kompatibilität. Deswegen werden wir die in WSFL definierten Web-Service-Prozessmodelle in ein abstraktes Modell auf Basis einer Prozessalgebra überführen und auf diesen Modellen mit Hilfe der formalen Semantik der Prozessalgebra die semantische Kompatibilität entscheiden.

Das Ziel dieser Arbeit ist das Finden eines geeigneten Kriteriums zum Nachweis der verklemmungsfreien Zusammenarbeit komponierter Web-Service-Prozessmodelle. Diese Arbeit versteht sich als ein erster Schritt in diese Richtung. Hierbei werden wir uns auf die sequenziellen und alternativen Kontrollstrukturen beschränken und zudem ausschließlich die Komposition zweier Web-Service-Prozessmodelle betrachten.

Im zweiten Kapitel werden wir die Begriffswelt vertiefen und auf die Problematik näher eingehen. Darauf aufbauend werden wir im dritten Kapitel eine adäquate Abbildung der Web-Service-Prozessmodelle in die Prozessalgebra CCS vorstellen und über die Beschreibung der Kombinierbarkeitseigenschaft zu der Definition des Kriteriums überleiten. Anschließend werden wir die Richtigkeit des Kriteriums beweisen und im fünften Kapitel abschließend einen Ausblick geben.

2 Web Services

In diesem Kapitel wollen wir die zugrundegelegte Begriffswelt darlegen, um darauf aufbauend das Problemfeld näher zu beleuchten. Zunächst werden wir auf die Motivation und die wirtschaftliche Relevanz der neuen Technologie Web Service eingehen.

2.1 Motivation

Aufbauend auf dem Internet als einheitliches Kommunikationsmedium entstehen globale Geschäftsprozesse durch Komposition lokaler Geschäftsprozesse. Diese lokalen Geschäftsprozesse werden in Autonomie entwickelt und dann über das Internet veröffentlicht. Hieraus ergeben sich aber nicht zu unterschätzende Anforderungen sowohl im betriebswirtschaftlichen als auch im technologischen Bereich. Da die IT-Welt stark heterogen ist und isolierte Lösungen bei dieser anspruchsvollen Zielsetzung nicht helfen, haben sich führende Softwareunternehmen die Aufgabe gestellt, eine entsprechende Technologie zur Verfügung zu stellen. Die nicht abschließend formal definierten Web Services werden wir nun in Anlehnung an die Stencil Group vorstellen [Gro01].

Web Services sind beliebig komplexe, sich selbst beschreibende, wieder verwendbare Softwarekomponenten, die Funktionalität zu logischen Einheiten kapseln und zugreifbar über standardisierte Internetprotokolle sind. Über die Protokolle können sie von Klienten gesucht und gefunden werden, sowie mit anderen Softwarekomponenten zu einem lose gekoppelten, verteilten System verbunden werden. In diesem Zusammenhang treten die Web Services als Bausteine auf, die ihrerseits selbst aus diesen Bausteilen bestehen können.

Auf der einen Seite werben bereits eine Vielzahl von Unternehmen mit dem Modewort Web Service. In dieser Euphorie werden Web Services als neue Generation verteilter Systeme gehandelt. Auf der anderen Seite gibt es viele Kritiker, die nur eine neue Marketingschlacht für eine Neuauflage bereits existierender Technologie sehen [vdA]. Zudem wird kritisiert, dass zur Zeit keine konsistente und standardisierte Terminologie verfügbar ist. Nach der Gartner Group [Gro02] haben Web Services bereits ihre höchste Aufmerksamkeit überschritten und befinden sich nun auf dem Weg der Konsolidierung, vgl. Abbildung 1. Dennoch wird kontrastiert, dass in spätestens fünf Jahren die Technologie ausgereift sein und sich im Markt etabliert haben wird.

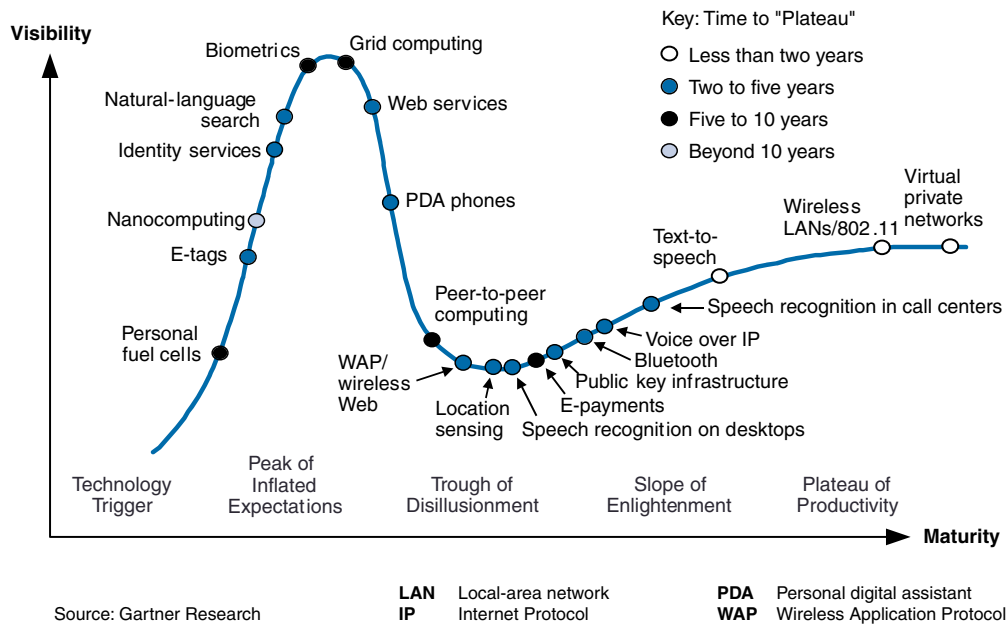


Abbildung 1: The 2002 Hype Cycle of Emerging Technologies

2.2 Problemstellung

Wie wir gerade dargestellt haben, ist der Web-Service-Ansatz eine sich entwickelnde Technologie, bei der es besonders auf die Spezifizierung sowohl der Technologie als auch auf die Modellierung von Web Services und deren Komposition ankommt. Die Modellierung als komplexe, Zeit und Geld intensive Komponente des Entwurfsprozesses soll in dieser Arbeit unterstützt werden. Hierzu werden wir ein Kriterium zur Entscheidung der semantischen Kompatibilität angeben. Nicht nur die hinreichende semantische sondern auch die notwendige syntaktische Kompatibilität ist für die Komponierbarkeit zweier Web-Service-Prozessmodelle entscheidend. Diesen Fakt wollen wir an einem Szenario aufzeigen.

Szenario: Zwei Menschen, die gerade die ersten Ansätze der englischen Sprache gelernt haben, wollen sich auf Englisch unterhalten. Beide sind sehr höflich und haben auch von den englischen Lords gehört, die ihr Gegenüber sehr zuvorkommend behandeln und die Antwort auf ihre Frage stets abwarten, bevor sie wieder aktiv das Gespräch fortführen. Das Gespräch beginnt und beide eröff-

nen aus ihrem Eifer heraus mit „Hallo, how are you?“und „Hallo, what are you doing?“. Durch die ungünstige Stimmenüberlagerung hat keiner der beiden etwas verstanden, da sie aber wie die Lords agieren wollen, warten beide auf die Antwort des Anderen. Leider ist es nun bei der Begrüßung zu einer Verklemmung gekommen.

Ein Nutzer wählt einen verfügbaren Web Service und komponiert diesen mit seinem Web Service, hierdurch entsteht ein verteiltes System. Wenn die Schnittstellen - im Szenario die Sprache Englisch - passen, entsteht ein syntaktisch korrektes, lose gekoppeltes System. Somit ist die syntaktische Kompatibilität erfüllt, dies kann einfach durch Parsen der Schnittstellen der beiden Web-Service-Prozessmodelle entschieden werden. Auf dieser Grundlage betrachten wir nun die semantische Kompatibilität. Ein syntaktisch kompatibles System muss kein verklemmungsfreies Verhalten haben, wie wir im Szenario gesehen haben. Deshalb ist die semantische Kompatibilität für unsere Problemstellung, der Entscheidung der Komponierbarkeit zweier Web-Service-Prozessmodelle, entscheidend. Das durch die Komposition zweier Web-Services-Prozessmodelle entstandene verteilte System kann eine Vielzahl möglicher Abläufe haben, wenn alle Abläufe keine Verklemmung aufweisen, dann nennen wir das System semantisch kompatibel. Für die Analyse der semantische Kompatibilität ist somit das Verhalten der Web Services, welches im Web-Service-Prozessmodell beschrieben wird, der Ansatzpunkt. Die Entscheidung ob semantische Kompatibilität vorliegt wird nicht durch Testen, sondern auf Basis der Web-Service-Prozessmodelle vorgenommen.

2.3 Akteure im Web-Service-Ansatz

Nachdem wir den Web-Service-Ansatz anhand eines abstrakten Szenarios betrachtet haben, wollen wir die Akteure und deren Zusammenspiel näher beleuchten (vgl. Abbildung 2).

Jeder Akteur ist entweder ein Nutzer, ein Anbieter oder ein Verwalter. Die Rolle des Anbieters übernimmt die Entwicklung eines Web Service, beschreibt diesen mit Schlüsselwörtern und wird diese Dienstleistung einem Verwalter anzeigen. Somit ist der Service veröffentlicht und kann von Dritten verwendet werden. Der Nutzer sucht bei Verwaltern nach einem für ihn geeigneten Web Service und muss hierzu die gewünscht Dienstleistung mit Schlüsselwörtern beschreiben. Hat der Nutzer einen entsprechenden Web Service gefunden so bin-

det er diesen Service an seinen Web Service und erzeugt dadurch einen verteilten Geschäftsprozess. Die Rolle des Verwalters hat in diesem Zusammenspiel ausschließlich die Funktion eines Verzeichnisdienstes für Web Services, der durch geeignete Katalogisierung den Kontakt zwischen dem Nutzer und Anbieter ermöglichen kann.

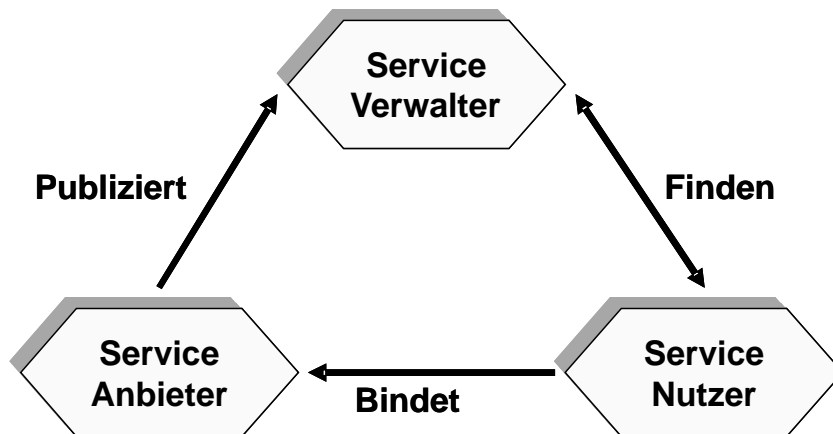


Abbildung 2: Rollen im Web-Service-Ansatz

2.4 Abstraktion auf WSFL

In diesem Gliederungspunkt werden wir die für die Betrachtung wesentlichen Eigenschaften von WSFL darstellen.

Die in WSFL beschriebenen Web-Service-Prozessmodelle sind in ihrer textuellen Form XML basiert und beschreiben das Verhalten der Web Services. In den Web-Service-Prozessmodellen werden azyklischen Graphen beschrieben, diese Graphen stellen die Kausalitäten der Aktivitäten dar. Dabei sind sowohl sequenzielle als auch alternative Kontrollstrukturen erlaubt, das heißt es dürfen entweder eine oder mehrere Aktivitäten von den Knoten ausgehen.

Web Services stellen für die Kommunikation gerichtete bidirektionale Nachrichtenkanäle, die mit Identifiern versehen sind, bereit. Im Web-Service-Prozessmodell wird zwischen intern und extern sichtbaren Aktivitäten unterschieden, für nähere Beschreibungen sei auf die WSFL-Spezifikation [Ley01] verwiesen. Die internen Aktivitäten, welche nicht der Kommunikation dienen, werden von nun an als τ -Aktivitäten bezeichnet, da die interne Berechnung von Ergebnissen keine neuen Erkenntnisse im Bezug auf unser Ziel der semantischen Kompatibilität bringen. Die τ -Aktivitäten bilden somit eine Klasse; die Gruppe der extern

sichtbaren Aktivitäten werden wir nun in zwei weitere Klassen unterteilen. Aktivitäten, die der Sendung von Nachrichten dienen, bilden eine Klasse und werden mit "!" gefolgt vom Identifier des Nachrichtenkanals bezeichnet, zum Beispiel !a. Entsprechend ist die dritte Klasse den Empfangsaktivitäten vorbehalten, sie werden mit "?" und dem Identifier des Nachrichtenkanals bezeichnet, ?a.

Bei der Kommunikation zweier Web Service muss erfüllt sein, dass zum Senden einer Nachricht der Empfänger zum Empfangen bereit ist, damit die beiden Web Services komponierbar sind. Dieses Paradigma wird auch als synchrone Kommunikation bezeichnet und stellt eine wesentliche Anforderung dar, die im Szenario aus Kapitel 2.2 nicht erfüllt wurde.

Nach diesen Ausführungen werden wir uns jetzt dem Lösungsansatz der semantischen Kompatibilität zuwenden. Als Ansatzpunkt werden wir das Web-Service-Prozessmodell verwenden, hierbei setzen wir die syntaktische Kompatibilität voraus.

3 Entwicklung des Kriteriums

In diesem Kapitel werden wir einen Ansatz zur Entscheidung der semantischen Kompatibilität vorstellen. Hierzu wird zunächst das vorgestellte Problem in ein abstraktes Modell auf Basis der Prozessalgebra CCS überführt, um anschließend ein Kriterium für die Kompatibilitätsprüfung vorzustellen.

3.1 Überführung von WSFL in Prozessalgebra

Zunächst stellt sich die Frage, warum wir Prozessalgebren als adäquates Mittel zur Lösung in Betracht ziehen. Zum einen müssen wir eine Überführung vornehmen, da zur Zeit keine Semantik für WSFL vorliegt und somit keine Analyse auf WSFL erfolgen kann. Zum anderen besitzt der gewählte weit verbreitete Vertreter *Calculus of Communicating Systems* (CCS) eine ähnliche Struktur. Im folgenden Abschnitt werden wir diesen Fakt näher erläutern und anschließend auf die Besonderheiten der Migration der Kommunikation zwischen Web Services eingehen.

3.1.1 Prozessalgebra CCS genügt der Betrachtung

Wie bereits erwähnt kann WSFL aufgrund der fehlenden Semantik nicht selbst analysiert werden. Deshalb haben wir eine Abbildung auf die Prozessalgebra CCS gewählt, unter anderem nicht nur weil die Strukturen im Bezug auf die Kommunikation sich ähneln, sondern auch die Nutzung der theoretischen Konzepte, die CCS unterstützt, für die Modellierung dieser interaktiven Systeme bedeutsam sind.

CCS ist für die Modellierung interaktiver Systeme entwickelt worden, anders als bei dem klassischen Automatenkonzept wird das Verhalten auch von außen über den Nachrichtenaustausch bestimmt. Die syntaktischen Konstrukte sind zum einen Aktivitäten, die durch Identifier dargestellt werden und zum anderen die Konnektoren für sequenzielle $;$, für alternative Verkettung $+$, sowie für parallele Ausführung der Aktivitäten $|$. Ein weiteres Instrument der Sprache CCS ist durch den Hiding-Operator gegeben $/$, mit dessen Hilfe können Aktivitäten zu einer synchronisierten Ausführung gekoppelt werden. Die durch diese Konstrukte gebildeten Terme werden als Agenten bezeichnet. Ein Agent stellt ein System dar, welches in der Lage ist über Kommunikation mit anderen Agenten zu interagieren.

Wir wollen von Daten in der Kommunikation abstrahieren, CCS abstrahiert grundlegend von Daten, während WSFL getypte Daten verwendet. Typkompatibilität ist eine Voraussetzung und kann im Vorfeld durch parsen leicht entschieden werden. Das Modell wird dadurch zwar allgemeiner aber verliert unter dem Gesichtspunkt der Kompatibilität nicht an Aussagekraft, da die Nachrichten als solche von Interesse sind aber der Nachrichteninhalt, also die für den Anwendungsfall spezifischen Daten, nicht. Getypte Daten liefern ebenso für die allgemeine Betrachtung der semantischen Kompatibilität keine zusätzliche Erkenntnisse. Zudem würde schon die Einbeziehung von ungetypten Daten eine Analyse durch die Zustandsexplosion nahezu unmöglich machen.

Die Mächtigkeit von CCS ist für unser Problem ausreichend. Wir wollen dies kurz anhand der wichtigen WSFL-Eigenschaften darstellen. Web Services sind endlich, Agenten in CCS sind es ebenso. Von WSFL betrachten wir sequenziell bzw. alternativ verknüpfte Aktivitäten, die wir in CCS durch die oben angeführten Konnektoren ausdrücken können. Die von uns eingeführte Abstraktion auf drei Klassen von Aktivitäten kann in CCS ohne Probleme übernommen werden, siehe Kapitel 3.1.3

In WSFL kann es mehrere Anfangszustände geben, wir wollen davon abstrahieren und eventuell mehrere Anfangszustände mittels vorgesetzter alternativer τ -Aktivität ohne Beschränkung zu einem Anfangszustand vereinen, dies stellt ausschließlich eine Erleichterung für das zu bildende Kriterium dar. WSFL beschreibt azyklische Graphen, diese sind problemlos überführbar. Ebenso problemlos ist die Annahme, dass zwei Agenten, die ausschließlich aus internen also τ -Aktivitäten bestehen, kompatibel sind.

Neben diesen Punkten hat CCS einen weiteren Vorteil, es ist technisch unterstützt. Die Universität von Edinburgh hat eine Workbench [CWB] veröffentlicht, die CCS analysieren kann. Zudem ist noch ein weiteres Tool namens daVinci [daV] für die Graphenanalyse verwandt worden. Die Überführung der einzelnen WSFL-Konstrukte, sowie die automatisierte Entscheidung der semantischen Kompatibilität kann unter [HS03] nachgelesen werden.

3.1.2 Kommunikation ist Synchronisation

Web Services kommunizieren laut WSFL-Spezifikation über bidirektionale Nachrichtenkanäle. Diese Kanäle können aber auch als zwei gerichtete und unidirektionale Nachrichtenkanäle dargestellt werden. Der Identifier des Nachrichtenka-

3 ENTWICKLUNG DES KRITERIUMS

nals entspricht dem Identifier der korrespondierenden Sende- und Empfangsaktivität.

Als zweiter Punkt ist anzumerken, dass die Kommunikation nur mit der Umgebung des Web Service stattfindet, somit schickt sich kein Web Service selbst eine Nachricht. Die Kommunikation ist ein sichtbares und damit beobachtbares Ereignis. Hierbei ist das Senden aktiv und das Empfangen passiv, eine erfolgreiche Kommunikation ist nur möglich wenn der Empfänger zum empfangen bereit ist ebenso wie der Sender, dann werden die beiden Aktivitäten zusammen ausgeführt. Damit ist durch das Warten auf die passive Empfangsbereitschaft eine Synchronisation in der Kommunikation enthalten.

3.1.3 Abbildung von Senden und Empfangen

Wie wir eben festgestellt haben besteht eine Asymetrie zwischen Senden und Empfangen. Dieser Unterschied muss in CCS ausgedrückt werden. Dem Senden ist eine aktive Rolle und dem Empfangen ist eine passive Rolle zugeordnet, das Senden ist also ein bestimmendes Ereignis und beruht auf einer Entscheidung für das Senden. Wir werden dies mit einer Sequenz zweier Aktivitäten darstellen, eine τ -Aktivität symbolisiert die Entscheidung zum Senden gefolgt von der Sendeaktivität. Das Empfangen wird als eine Aktivität aufgefaßt.

Die Asymmetrie zwischen Senden und Empfangen werden wir in zwei Teilschritten nach CCS übertragen. Der erste Schritt ist die dargestellte Überführung des Sendens in zwei Aktivitäten. Der zweite Schritt ist nötig, damit die korrespondierenden Sende- und Empfangsaktivitäten auch zusammen ausgeführt werden. Dazu werden die Identifier der korrespondierenden Aktivitäten mit dem Hiding-Operator vermerkt. Die folgende Tabell stellt die Abbildung für einen Austausch der Nachricht a dar.

WSFL-Abstraktion	Überführung	CCS
Sendeaktivität	$\tau!a$	τa
Empfangsaktivität	$?a$	$'a$
interne Aktivität	τ	τ

Bei dieser Modellierung bleibt die gerichtete synchrone Kommunikation zwischen zwei Web Services erhalten, zudem ist es uns gelungen von der fehlenden Semantik zu abstrahieren. Auf dieser Basis können wir nun das Problem der semantischen Kompatibilität analysieren.

3.2 Kombinierbarkeitseigenschaft

In diesem Abschnitt werden wir jetzt das betrachtete Problem der semantischen Kompatibilität als Eigenschaft der Komposition zweier Agenten formulieren.

Unter der Kombinierbarkeitseigenschaft verstehen wir die *verklemmungsfreie* Zusammenarbeit zweier komponierter Agenten, das heißt alle Abläufe des komponierten Systems müssen in wohl definierten Endzuständen enden.

Um die betrachtete Eigenschaft näher darzustellen und eine Intuition im Umgang mit dem Problemfeld zu entwickeln, werden wir zunächst die Verklemmung näher beschreiben und anschließend die Anforderungen an das zu bildende Kriterium formulieren.

3.2.1 Kombinierbarkeit von Agenten

Das Transitionssystem eines Agenten kann, wie wir oben gezeigt haben, als CCS-Term und dieser ebenso als Graph dargestellt werden. Exemplarisch ist der CCS-Term $((\tau.!a.?b)+(\tau.!a.?c))$ in der Abbildung 3 als Graph dargestellt. Der Graph besteht aus nummerierten Knoten und Kanten, welche die Aktivitäten des Agenten repräsentieren. Im Zustand 1 besteht eine Alternative repräsentiert durch die beiden ausgehenden Kanten, von diesen Kanten wird eine nichtdeterministische gewählt, dies stellt ein exklusives Oder dar.

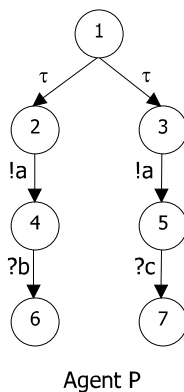


Abbildung 3: Graph eines Agenten

Bei der Komposition zweier Agenten kommt es uns auf die Verklemmungsfreiheit aller Abläufe an. Die folgenden zwei Beispiele sind aus dem Buch *Communication and Concurrency* entnommen [Mil89] und sollen dem Leser einen Eindruck in die zu beachtenden Probleme bei der Kombinierbarkeit geben.

3 ENTWICKLUNG DES KRITERIUMS

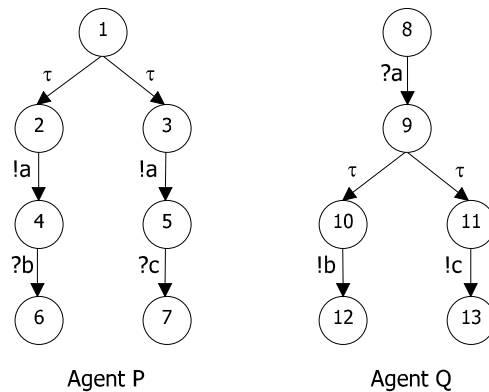


Abbildung 4: verklemmendes komponentiertes System

Abbildung 4 zeigt die Komposition zweier Agenten. Agent P und Q stellen zusammen ein komponentiertes System dar, das im verteilten Zustand (1,8) startet. Ein Verhalten dieser Komposition wollen wir nun kurz beschreiben. Wenn Agent P in den Zustand 2 wechselt und anschließend ein a sendet, so gehen beide Agenten vom verteilten Zustand (2,8) in den verteilten Zustand (4,9) über, da die Sende- und Empfangsaktivitäten zusammen ausgeführt werden. Agent Q entscheidet sich nun c zu senden und befindet sich jetzt im Zustand 11, der verteilte Zustand ist somit (4,11). Da c gesendet werden soll, muss Agent P empfangsbereit für c sein, dies ist im erreichten verteilten Zustand nicht möglich und das komponentierte System ist in diesem Ablauf verklemmt. Daraus folgt, dass die beiden Agenten nicht verklemmungsfrei zusammenarbeiten können und somit das System nicht kombinierbar ist.

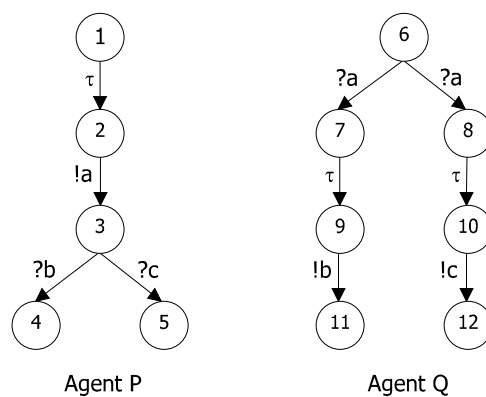


Abbildung 5: verklemmungsfreies komponentiertes System

Das Beispiel aus Abbildung 5 nennen wir kompatibel, Agent P und Agent Q

sind also kombinierbar, da jeder Ablauf in einem verteilten Endzustand endet, also keine Verklemmung aufgetreten ist. Anzumerken ist, dass die jeweils gleichnamigen Agenten aus den Abbildungen 4 und 5 zwar die selben Aktivitäten in der gleichen Reihenfolge aufweisen, aber durch andere kausale Zusammenhänge auch verklemmende komponierte Systeme entstehen. Oft ist nicht durch bloßes Hinsehen, wie bei diesen kleinen Beispielen zu entscheiden, ob die beiden Agenten kombinierbar sind. Aufbauend auf diesen Erkenntnissen werden wir im Folgenden die Anforderungen für Kombinierbarkeit näher beleuchten und anschließend das Kriterium definieren.

3.2.2 Anforderungen

Während wir nun die Anforderungen für Kombinierbarkeit an ein System formulieren, wollen wir zugleich *intuitiv* Begriffe für die Definition des Kriteriums bilden. Als Ausgangspunkt stellt sich das komponierte System dar, als notwendig wird von uns die syntaktische Kompatibilität erachtet, hinreichend ist zudem eine verklemmungsfreie Zusammenarbeit der beiden Agenten. Das bedeutet, alle Abläufe des komponierten Systems müssen in Endzuständen enden. Die Untersuchung der Abläufe ist endlich, da die Agenten endlich sind und somit endlich viele endliche Abläufe entstehen.

Ein *Agent* ist ein Transitionssystem, welches als Graph dargestellt werden kann. Die zu untersuchenden Abläufe des komponierten Systems entstehen bei der Ausführung der beiden Agenten, Ausführung meint die Abarbeitung der Agenten und die dabei auftretende Kommunikation.

Jeder Agent besteht aus Zuständen und Überführungen zwischen diesen Zuständen. Ein Agent hat nach unserer Annahme einen Anfangszustand und möglicherweise mehr als einen Endzustand. Bei der Komposition werden wir zur besseren Verständlichkeit von *verteilten Zuständen* reden, da hierdurch die Unterscheidung zwischen der Betrachtung eines Agenten bzw. der Komposition erleichtert wird. Durch die Komposition ergibt sich also ein verteilter Anfangszustand und eine Menge von verteilten Endzuständen, die mindestens einelementig ist.

Ein verteilter Zustand wird in einen verteilten Zustand durch eine *verteilte Aktivität* überführt, mittels der Ausführung einer lokalen τ -Aktivität bzw. beide Agenten führen zusammen zwei *korrespondierende Aktivitäten* aus, das heißt Agent P führt eine Sendeaktivität und Agent Q führt die Empfangsaktivität

mit dem selben Identifier aus.

Um das Problem der semantischen Kompatibilität vollständig zu erfassen, müssen von einem verteilten Zustand aus die zukünftig ausführbaren Sende- und Empfangsaktivitäten gegenübergestellt werden. Da das Senden aktiv ist, muss auf alles, was gesendet werden kann, reagiert werden können, das heißt der Empfang muss möglich sein. Wenn wir bei der Gegenüberstellung nur den verteilten Anfangszustand betrachten, wird das Problem jedoch nicht vollständig gelöst, da die Mengen der beiden Aktivitätsklassen zwar in der gewünschten Teilmengenbeziehung stehen, aber noch nicht jeder mögliche kausale Zusammenhang abgedeckt wird, siehe Abbildung 4. Daraus folgt, wir müssen jeden erreichbaren verteilten Zustand auf Kompatibilität prüfen. Hierdurch ergibt sich, dass die jeweilige Prüfung nur die Aktivitäten mit einzubeziehen braucht, die von den jeweiligen verteilten Zuständen durch eventuell vorhandene τ -Pfade erreicht werden können. Also braucht man keine Aktivitäten zu betrachten, die nach der ersten Sende- bzw. Empfangsaktivität auftreten.

Ein *verteilter Zustand bezeichnen wir als kompatibel*, wenn es keine durch eventuell vorhandene τ -Pfade erreichbaren Sende- bzw. Empfangsaktivitäten gibt. Ebenso ist der verteilte Zustand kompatibel, wenn zu alle durch eventuell vorhandene τ -Pfade erreichbaren Sendeaktivitäten der andere Agent eine korrespondierenden Empfangsaktivität, die ebenfalls durch einen eventuell vorhandenen τ -Pfad erreicht werden kann, aufweist. Dabei muss allerdings erfüllt sein, dass nur ein Agent in dem betrachteten Zustand Sendeaktivitäten aufweist, sonst könnten sich beide Agenten für das Senden entscheiden und das System würde verklemmen. Zusätzliche Empfangsaktivitäten sind dagegen nicht einzuschränken, aufgrund ihrer passiven Rolle.

Wir müssen also zusammenfassend verlangen, dass jeder mögliche Ablauf von dem verteilten Anfangszustand aus in einem verteilten Endzustand endet, dabei ist jeder erreichte verteilte Zustand kompatibel, dann sind die *beiden Agenten kombinierbar*.

3.3 Definition des Kombinierbarkeitskriteriums

Auf Basis der dargestellten Anforderungen werden wir das syntaktische Kriterium formulieren. Als Grundlage der Definition des Kombinierbarkeitskriteriums geben wir zunächst die Definition für einen Agenten an, um auf dieser Basis die Kompatibilität von verteilten Zuständen zu formulieren und anschließend die

Kombinierbarkeit zweier Agenten zu definieren.

Definition 1 (Agent):

Ein Agent P ist ein Graph $P=(Z_P, K_P)$, Z_P ist die Menge der Zustände und K_P die Menge der Kanten. Die Kanten $[p,q]$, $p,q \in Z_P$, sind die Aktivitäten, das bedeutet das Label einer Kante $\text{label}([p,q])$ ist entweder eine τ -Aktivität, eine Sendeaktivität $!\ll\text{Identifer eines Nachrichtenkanals}\gg$ oder eine Empfangsaktivität $?\ll\text{Identifer eines Nachrichtenkanals}\gg$. *

Lemma 1:

Sei die extern sichtbare Aktivität a eine Sendeaktivität $!a$, dann ist durch die Überführung nach CCS eine nicht verzweigende Abfolge von τ und $!a$ entstanden. *

Das Lemma stellt eine wesentliche Eigenschaft der in Kapitel 3.1.3 dargestellten Überführung des Sendens heraus. Bei unser Abstraktion auf die Entscheidung für das Senden, modelliert durch eine τ -Aktivität, kann nach dieser Entscheidung keine Alternative zum Senden zur Verfügung stehen. Da sonst keine eindeutige Entscheidung für den aktiven Vorgang des Sendens vorliegen könnte.

Um die Kombinierbarkeit zweier Agenten zu definiern, müssen wir, wie in den Anforderungen dargestellt, die Kompatibilität verteilter Zustände charakterisieren. Dazu folgen zuerst die Definition eines verteilten Zustandes, gefolgt von den Mengendefinitionen für Sende und Empfangsaktivitäten. Diese sind für die Feststellung der Kompatibilität zwingend, wie bereits erwähnt werden diese Mengen von einem Zustand im Agenten gebildet und umfassen die jeweils ersten Sende- bzw. Empfangsaktivitäten der von diesem Zustand ausgehenden Pfade. Hierbei kann eine Sequenz von τ -Aktivitäten den Sende- und Empfangsaktivitäten vorgelagert sein, um dies zu erfassen wird ein τ -Pfad definiert.

Definition 2 (verteilter Zustand):

Ein verteilter Zustand (p,q) ist ein Tupel von Zuständen zweier Agenten P und Q , mit $p \in Z_P$ $q \in Z_Q$. *

Definition 3 (τ -Pfad):

Ein τ -Pfad ist ein Pfad der von einem Zustand p zum Zustand r im Agenten P durch eine Sequenz von Kanten $[p,q_1], \dots, [q_n,r]$, $n \in \mathcal{N}$ und $n \geq 1$ mit $\text{label}([p,q_1])=\tau, \dots, \text{label}([q_n,r])=\tau$ gebildet wird. *

Im folgenden werden die Definitionen für die Mengenbildung der Sende-

und Empfangsaktivitäten angegeben. Diese Mengen werden für die Kompatibilitätsprüfung der verteilten Zustände benötigt, sie werden von einem Zustand im Graphen gebildet. Dabei werden von diesem Zustand ausgehend die weiteren Pfade untersucht, jeweils bis zu der ersten Kante mit dem Label einer Sende- bzw. Empfangsaktivität.

Definition 4 (Sendeaktivitätsfunktion $s(p)$):

Sei p ein Zustand im Graphen P und sei $!x$ eine Sendeaktivität, $!x$ ist in der Menge $s(p)$, wenn entweder eine Kante $[p,q] \in K_P$ mit $\text{label}([p,q]) = !x$ existiert oder ein Zustand r und ein τ -Pfad von p nach r und eine Kante $[r,q] \in K_P$ mit $\text{label}([r,q]) = !x$ existiert. *

Definition 5 (Empfangsaktivitätsfunktion $r(p)$):

Sei p ein Zustand im Graphen P und sei $?x$ eine Empfangsaktivität, $?x$ ist in der Menge $r(p)$, wenn entweder eine Kante $[p,q] \in K_P$ mit $\text{label}([p,q]) = ?x$ existiert oder ein Zustand r und ein τ -Pfad von p nach r und eine Kante $[r,q] \in K_P$ mit $\text{label}([r,q]) = ?x$ existiert. *

Wir haben die zur Kommunikation dienenden Aktivitäten in zwei Mengen aufgeteilt und können nun auf dieser Basis die Kompatibilität eines verteilten Zustandes als Relation komp_a definieren. Die Definition ist in drei logische Teile untergliedert, zum einen wird als Sicherheitscharakteristik verlangt, dass alles was gesendet werden kann empfangbar sein muss. Zum anderen wird verlangt, dass genau einer der beiden Agenten sendet oder drittens sich jeder Agenten in einem Endzustand befindet, dies spiegelt den Charakter der Lebendigkeit wieder.

Definition 6 (Relation komp_a):

Der Zustand p des Agenten P und der Zustand q des Agenten Q sind in der Relation $\text{komp}_a(p,q)$, wenn:

- $s(p) \subseteq r(q) \wedge s(q) \subseteq r(p)$
- $((s(p) = \emptyset \vee s(q) = \emptyset) \wedge (\neg s(p) = s(q) = \emptyset))$
 \vee
 $((r(p) \cup s(p) = \emptyset) \iff (r(q) \cup s(q) = \emptyset))$

Der verteilte Zustand (p,q) heißt dann kompatibel. *

Nachdem wir die Kompatibilität von verteilten Zuständen dargestellt haben, müssen wir das Zusammenspiel der beiden Agenten näher betrachten. Hierbei

müssen wir im besonderen den Unterschied zwischen der lokalen Ausführung einer τ -Aktivität und der zusammen stattfindenden Ausführung korrespondierender Aktivitäten spezifizieren. Ebenso müssen wir grundlegend die Ausführung von Aktivität definieren, dazu werden wir den Begriff der verteilten Aktivität verwenden.

Definition 7 (korrespondierende Aktivitäten):

Eine Sendeaktivität heißt korrespondierend zu einer Empfangsaktivität und umgekehrt, wenn sie den selben Identifier eines Nachrichtenkanals tragen. *

Definition 8 (verteilte Aktivität):

Durch eine verteilte Aktivität wird ein kompatibler verteilter Zustand (p,q) der Agenten P und Q in einen verteilten Zustand (p',q') überführt, wenn eine der folgenden Aussagen gilt:

- es existiert eine Kante der Form $[p,p'] \in K_P$ mit $\text{label}([p,p']) = \tau$ bzw. $[q,q'] \in K_Q$ mit $\text{label}([q,q']) = \tau$
- es existiert eine Kante der Form $[p,p'] \in K_P$ und eine Kante der Form $[q,q'] \in K_Q$ und die Label der Kanten sind korrespondierende Aktivitäten *

Wir haben die verteilte Aktivität nur auf *kompatiblen* verteilten Zuständen definiert, dies reicht für unsere Zielsetzung aus. Da von einem nicht kompatiblen verteilten Zustand keine weitere Untersuchung nötig ist und so auch keine verteilte Aktivität mehr ausgeführt werden muss. Mit den vorhergehenden Definitionen können wir nun abschließend die Kombinierbarkeit zweier Agenten definieren.

Definition 9 (Relation kombi):

Die Agenten P und Q sind in der Relation $\text{kombi}(P,Q)$, wenn folgendes gilt:

- die Startzustände p_0, q_0 der Agenten P und Q bilden einen kompatiblen verteilten Zustand (p_0,q_0)
- aus jedem erreichten kompatiblen verteilten Zustand (r,s) wird durch jede mögliche Ausführung von verteilter Aktivität ein verteilter Zustand (r',s') , der $\text{kombi}(r',s')$ erfüllen muss, erreicht

Die Agenten P und Q heißen dann kombinierbar. *

4 Beweis

Wir haben ein Kriterium für die betrachtete Eigenschaft der semantischen Komposition zweier Agenten gefunden. Um dieses einer gesicherten Anwendung zuzuführen, müssen wir die Richtigkeit des Kriteriums nachweisen. Hierzu stellen wir ein Theorem auf, welches die Eigenschaft in Beziehung zu dem Kriterium stellt, darauf aufbauend werden wir die Korrektheit und die Vollständigkeit beweisen.

Für das Theorem benötigen wir zwei formale Definitionen, zum einen die Relation verklemmungsfrei, sie definiert die verklemmungsfreie Zusammenarbeit zweier Agenten, und zum anderen für diese Relation den Begriff des Ablaufes.

Definition 10 (Ablauf):

Ein Ablauf ist eine maximale Sequenz von verteilter Aktivität. *

Definition 11 (Reaktion verklemmungsfrei):

Wenn jeder Ablauf im komponierten System der Agenten P, Q vom verteilten Anfangszustand aus in einem verteilten Endzustand, in dem kein Agent eine Aktion aufweist, endet, dann sind die Agenten P, Q in der Relation verklemmungsfrei(P, Q). Die Agenten P, Q heißen dann verklemmungsfrei. *

Theorem 1:

Die Agenten P und Q sind in der Relation verklemmungsfrei genau dann wenn beide Agenten in der Relation kombi sind. *

4.1 Korrektheit (\Leftarrow)

Die Voraussetzung besagt, dass die beiden Agenten P und Q kombinierbar sind, kombi(P, Q). Nach der Definition von kombi sind die erreichbaren verteilten Zustände des komponierten Systems kompatibel. Zunächst werden wir beweisen, dass aus jedem dieser kompatiblen verteilten Zustände verteilte Aktivität ausgeführt werden kann oder ein Endzustand vorliegt.

Lemma 2:

Jeder verteilte Zustand (p, q) , der die Reaktion kompa(p, q) erfüllt, ist Endzustand oder mindestens eine verteilte Aktivität kann ausgeführt werden. *

Das Lemma sagt einen Fortgang im verteilten System voraus. Um das Lemma zu beweisen, müssen wir mit den zwei Fällen von verteilter Aktivität und den

Möglichkeiten, wie ein kompatibler verteilter Zustand beschaffen ist, argumentieren.

Beweis von Lemma 2:

Die Voraussetzung ist ein verteilter Zustand (p,q) der kompatibel ist. Wir können drei mögliche Ausgestaltungen von derartigen Zuständen nach Definition von kompa unterscheiden.

- 1. Fall:** Im Zustand p und im Zustand q existieren keine Kanten, damit ist ein Endzustand erreicht.
- 2. Fall:** Mindestens in einem Zustand existiert eine Kante mit einem τ -Label, nach Definition kann eine verteilte Aktivität ausgeführt werden.
- 3. Fall:** Genau ein Zustand sei dieser obda. p weist mindestens eine Kante der Form $[p,r]$ mit $\text{label}([p,r])=!x$ auf, dann muss nach Definition von kompa der Zustand q entweder eine Kante $[q,r]$ mit $\text{label}([q,r])=?x$ oder einen τ -Pfad von Zustand q zum Zustand t mit einer Kante $[t,u]$ und $\text{label}([t,u])=?x$ aufweisen. Dann ist nach Definition eine verteilte Aktivität ausführbar. \square

Wir haben gesehen, dass aus jedem kompatiblen verteilten Zustand verteilte Aktivität ausgeführt werden kann oder ein Endzustand vorliegt. Aus diesem Zusammenhang und der Eigenschaft der Endlichkeit der betrachteten Agenten folgt, dass im komponierten System nur endliche Abläufe existieren, die ausschließlich aus kompatiblen verteilten Zuständen bestehen. Da Abläufe maximal sind, gelangen wir von dem kompatiblen verteilten Anfangszustand immer in einen kompatiblen verteilten Zustand, der keine Kante ausweist, also ein Endzustand ist. Damit gilt $\text{verklemmungsfrei}(P,Q)$. \square

4.2 Vollständigkeit (\Rightarrow)

Die Voraussetzung ist $\text{verklemmungsfrei}(P,Q)$, um die Vollständigkeit nachzuweisen, werden wir einen Widerspruchsbeweis führen. Unsere Annahme ist, aus $\text{verklemmungsfrei}(P,Q)$ folgt, dass die Agenten P und Q nicht kombinierbar sind. Das bedeutet, es muss mindestens ein verteilten Zustand in den möglichen Abläufen des komponierten Systems geben, der nicht kompatibel ist. Wir müssen also die logischen Teile der Definition von kompa untersuchen:

- 1. Fall:** Eine Sendeaktivitätsmenge ist keine Teilmenge der entsprechenden Empfangsaktivitätsmenge. Obda. betrachten wir den Fall:

- Agent P : $(\tau.!a+...)$

- Agent Q: (...), Agent Q kann a nicht empfangen
- Der verteilte Anfangszustand ist nicht kompatibel.
- Nach Lemma 1 kann keine Alternative nach der Ausführung der verteilten Aktivität τ , also der Entscheidung zu Senden von a, im Agent P vorhanden sein. Somit ist ein verklemmender Zustand erreicht, da Agent P ein a senden will aber Agent Q nicht für den Empfang bereit ist. Widerspruch zur Annahme, da verklemmungsfrei nicht gilt.

2. Fall: Beide Sendeaktivitätsmengen sind nicht leer. Obda. betrachten wir den Fall:

- Agent P: $(\tau.!a+?b+...)$
- Agent Q: $(\tau.!b+?a+...)$
- Der verteilte Anfangszustand ist nicht kompatibel.
- Nach Lemma 1 kann keine Alternative nach den Ausführungen der verteilten Aktivitäten τ , also der Entscheidung zu Senden von a und b, im Agent P und Q vorhanden sein. Somit ist ein verklemmender Zustand erreicht, da Agent P nicht b empfangen und Agent Q nicht a empfangen kann, aber beide Agenten jeweils die entsprechende Nachricht senden wollen. Widerspruch zur Annahme, da verklemmungsfrei nicht gilt.

3. Fall: Ein Endzustand ist nicht immer erreichbar. Obda. betrachten wir den Fall:

- Agent P: $(\tau.?a+...)$
- Agent Q: (...), Agent Q kann a nicht senden
- Der verteilte Anfangszustand ist nicht kompatibel.
- Es wird ein verklemmender Zustand erreicht, nach der Ausführung der verteilten Aktivität τ im Agenten P, da Agent Q die Nachricht a nicht senden kann aber Agent P darauf wartet. Widerspruch zur Annahme, da verklemmungsfrei nicht gilt.

Wir haben in allen Fällen einen Widerspruch gezeigt, daraus folgt, dass unsere Annahme falsch ist. Somit folgt aus $\text{verklemmungsfrei}(P,Q)$, dass beide Agenten kombinierbar sind, $\text{kombi}(P,Q)$ gilt. \square

5 Zusammenfassung und Ausblick

Die mit dieser Arbeit verfolgte Zielsetzung, der Bereitstellung eines Kriteriums für die semantische Kompatibilität, wurde unter dem Gesichtspunkt der Einschränkung auf die Komposition zweier Web Services, die ausschließlich mit alternativen und sequenziellen Aktivitätsverknüpfungen arbeiten, erfüllt. Um dieses Ziel zu erreichen mussten wir aufgrund der fehlenden formalen Semantik von WSFL die Web-Service-Prozessmodelle in abstrakte aber analysierbare Modelle auf Basis von CCS überführen. Auf diesen CCS Agenten konnten wir dann ein Kriterium für die semantische Kompatibilität definieren und es durch die Beweisführung einer gesicherten Anwendung zuführen. Dies stellt ein ermutigendes Ergebnis für weiterführende Untersuchungen dar, die in der Bearbeitung parallel ablaufender Aktivitäten, sowie der Äquivalenz zweier Agenten liegen können. Der Äquivalenzbegriff hat nicht nur einen theoretischen Wert sondern auch einen praktischen Sinn. Zum Beispiel stellen wir uns eine Situation vor, in der ein Nutzer einen Web Service durch einen neueren Web Service ersetzen möchte. Dabei ist fraglich, ob der neue Web Service die selbe Spezifikation erfüllt, also beide ununterscheidbar sind, das heißt äquivalent sind.

Neben diesen Ansätzen unterliegt die Technologie einer ständigen Weiterentwicklung. Zum einen ist der Quality of Service zu nennen. Inbegriffen sind die Sicherheits- und Authentifikationskomponenten, die Komponenten zur Koordination von Abläufen sowie die Transaktionskontrolle. Dies ist im besonderen für die Sicherheit von internetbasierten Systemen relevant, aber auch für den immer stärker werdenden Trend zu innerbetrieblichen Anwendungen. In dem vor allem die Probleme der Transaktionskontrolle eine übergeordnete Rolle im Vergleich zu Übertragungszeiten und asynchronen Protokollen spielen. Zum anderen werden so genannte Business Grids vorgesehen, mit denen ein Anwender nicht mehr einen speziellen Partner sucht, sondern eine Funktionalität nachfragt.

Einen weiteren Schritt in die Richtung der Standardisierung und Durchsetzung der Technologie haben IBM, Microsoft und BEA vollzogen. WSFL wurde in dieser Gemeinschaft mit den Konkurrenzsprache XLANG von Microsoft zu BEPEL4WS vereint. Durch diese Gemeinschaft wird die weitere Verbreitung von Web Services gestärkt, zwar ist die Entwicklung noch in einem ständigen Fluss, aber es bestehen gute Chancen für die Durchsetzung am Markt.

Literatur

- [CWB] Edinburgh Concurrency Workbench. Available from <http://www.dcs.ed.ac.uk/home/cwb/>.
- [daV] daVinci v2.1. Available from <http://www.informatik.uni-bremen.de/daVinci>.
- [Gro01] The Stencil Group. Defining Web Services. Technical report, 2001. Available from <http://www.stencilgroup.com>.
- [Gro02] Gartner Group. The Monthly Research Review. Technical report, Juni 2002. Available from <http://www.gartner.com>.
- [HS03] Dirk Hain and Christian Stahl. Studienarbeit: Komposition von Web Services, 2003.
- [Ley01] Prof. Dr. Frank Leymann. Web Services Flow Language (WSFL 1.0). Technical report, IBM Software Group, May 2001. Available from <http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>.
- [Mil89] R. Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.
- [vdA] W.M.P. van der Aalst. Dont go with the flow: Web services composition standards exposed. Web Services - Been there done that?, Trends and Controversies. Jan/Feb 2003 issue of IEEE Intelligent Systems (to appear).
- [vG01] R.J. van Glabbeek. The Linear Time - Branching Time Spectrum I. The Semantics of Concrete, Sequential Processes. Technical report, Computer Science Department, Stanford University, 2001. in Handbook of Process Algebra.